US006766516B1

(12) **United States Patent**
Draganic

(10) **Patent No.: US 6,766,516 B1**
(45) **Date of Patent: *Jul. 20, 2004**

(54) **PROCESSOR SHARING TECHNIQUE FOR COMMUNICATIONS AND OTHER DATA PROCESSING ON A SAME PROCESSOR**

(75) Inventor: **Zarko Draganic**, Menlo Park, CA (US)

(73) Assignee: **Altocom, Inc.**, Irvine, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/448,649**

(22) Filed: **Nov. 24, 1999**

Related U.S. Application Data

(63) Continuation of application No. 08/780,611, filed on Jan. 8, 1997, now Pat. No. 5,995,540.

(51) **Int. Cl.**[7] **G06F 9/46**; H04B 1/38
(52) **U.S. Cl.** **718/102**; 375/222
(58) **Field of Search** 379/93.35, 215.01, 379/93.01, 93.26–93.34, 35, 93, 215; 375/222; 710/8, 14, 20, 62; 709/100–108

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,029,912	A	*	6/1977	Geigel et al.	379/406.07
4,943,942	A	*	7/1990	Dunnion	375/222
4,951,309	A	*	8/1990	Gross et al.	379/102.04
4,964,120	A	*	10/1990	Mostashari	370/228
4,965,641	A	*	10/1990	Blackwell et al.	375/219
5,491,721	A	*	2/1996	Cornelius et al.	375/222
5,982,814	A	*	11/1999	Yeh et al.	375/222
5,995,540	A	*	11/1999	Draganic	375/222
5,999,526	A	*	12/1999	Garland et al.	370/352
6,333,974	B1	*	12/2001	Liang et al.	379/93.35

OTHER PUBLICATIONS

ITU–T Draft Recommendation G.992.2, Transmission Systems and Media, "Splitterless Asymmetric Digital Subscriber Line (ADSL) Transceivers,"International Telecommunication Union, Feb. 17, 1999, pp. 3–153.

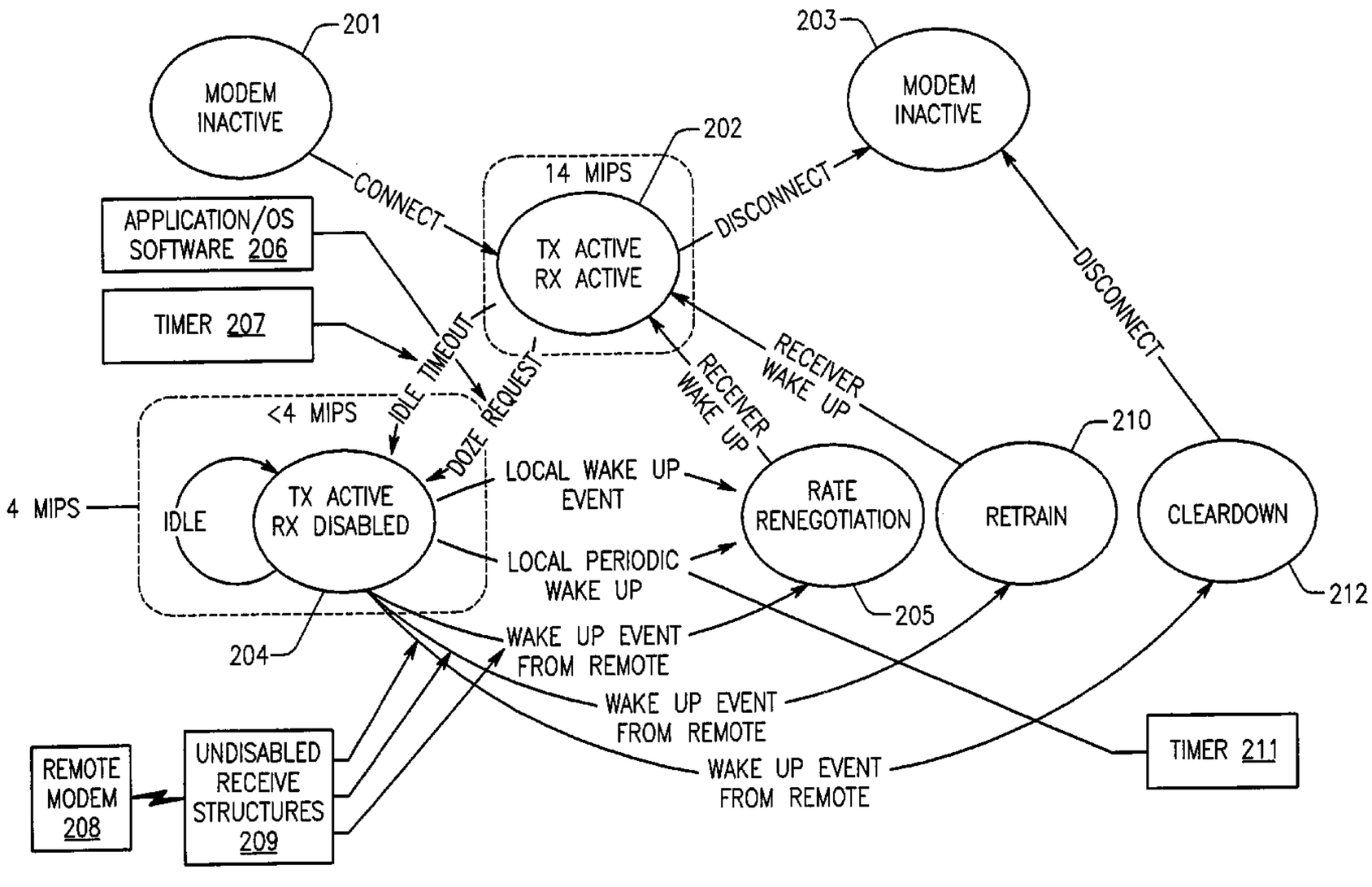
* cited by examiner

Primary Examiner—Meng-Al T. An
Assistant Examiner—Syed Ali
(74) *Attorney, Agent, or Firm*—Zagorin O'Brien & Graham LLP

(57) **ABSTRACT**

The computational load imposed by communications software executed on a general purpose processor can be significantly reduced by exploiting periods during an active connection when no data is being received. In particular, execution of many receive path signal processing algorithms can be disabled when no data is being received. The transmit path continues output modulation as with a normal connection, so as to trick a remote communications device into believing the connection is still normal. However, substantial portions of the local receive path can be disabled, thereby reducing computational load on the general purpose processor and freeing additional compute cycles for application and/or operating system program use.

12 Claims, 5 Drawing Sheets



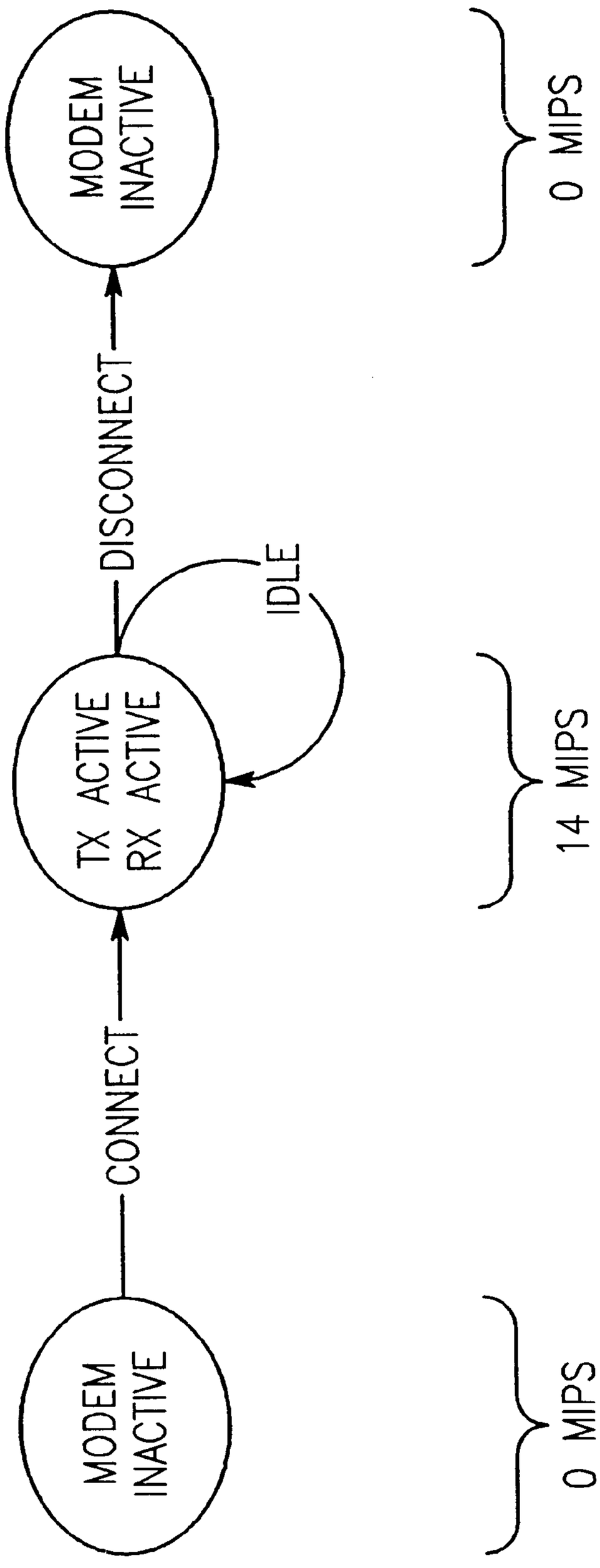


FIG. 1 (PRIOR ART)

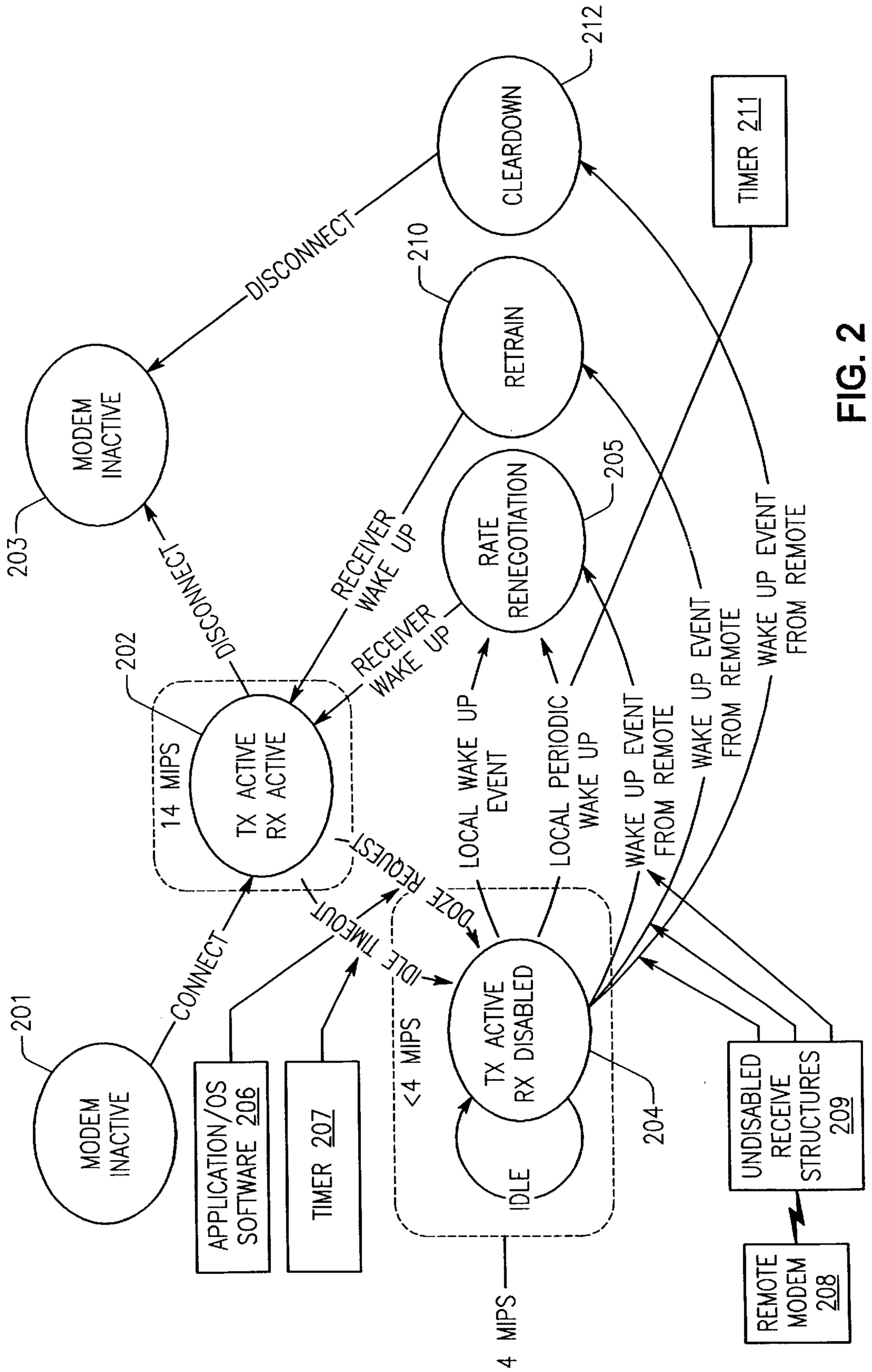

MODEM INACTIVE
203
DISCONNECT
CLEARDOWN
212
TIMER 211
210
RETRAIN
RECEIVER WAKE UP
RECEIVER WAKE UP
RATE RENEGOTIATION
205
202
14 MIPS
TX ACTIVE RX ACTIVE
DISCONNECT
LOCAL WAKE UP EVENT
LOCAL PERIODIC WAKE UP
WAKE UP EVENT FROM REMOTE
WAKE UP EVENT FROM REMOTE
WAKE UP EVENT FROM REMOTE
DOZE REQUEST
IDLE TIMEOUT
CONNECT
201
MODEM INACTIVE
APPLICATION/OS SOFTWARE 206
TIMER 207
<4 MIPS
TX ACTIVE RX DISABLED
IDLE
204
4 MIPS
UNDISABLED RECEIVE STRUCTURES 209
REMOTE MODEM 208

FIG. 2

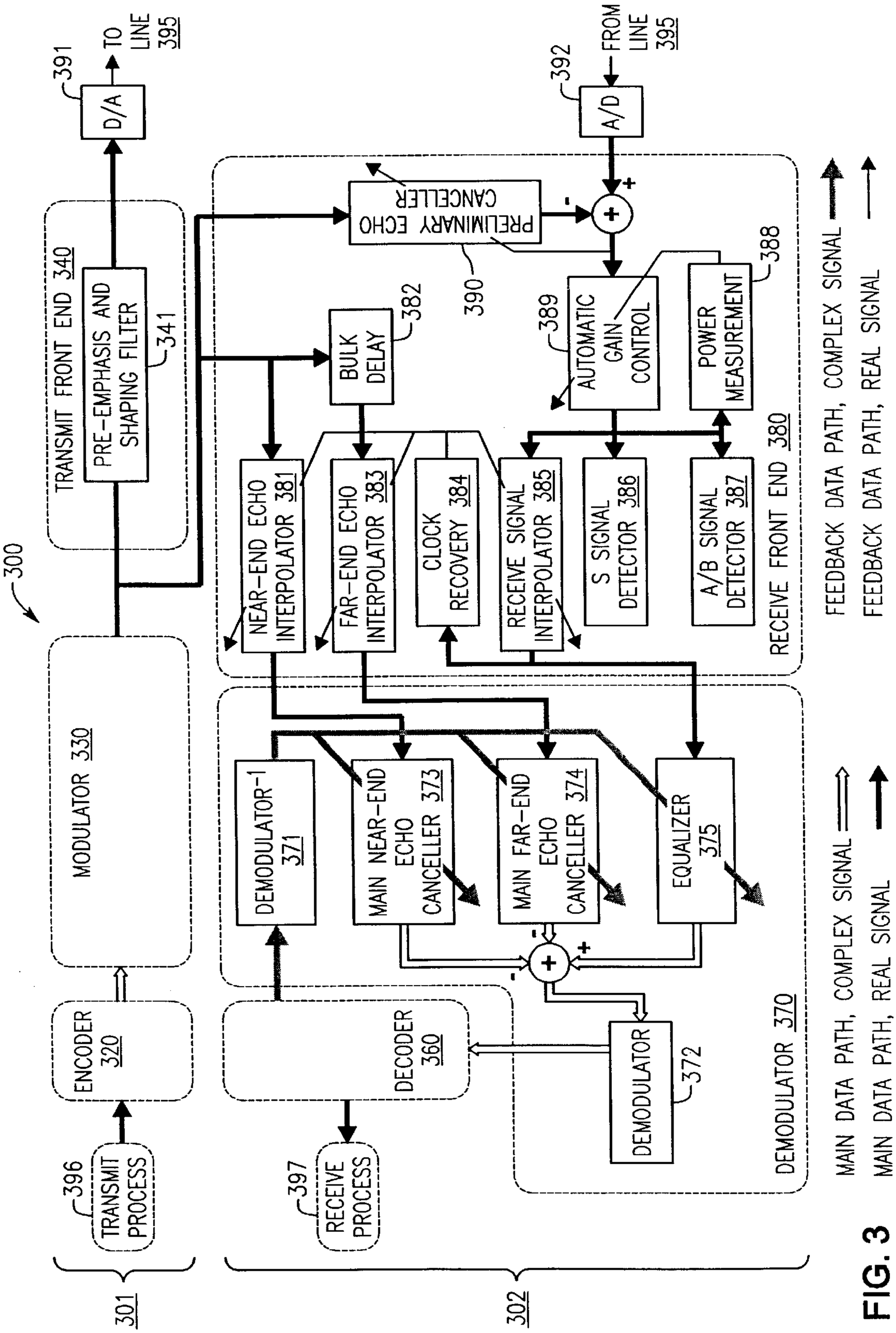

300
TRANSMIT FRONT END 340
PRE-EMPHASIS AND SHAPING FILTER 341
D/A 391
TO LINE 395
MODULATOR 330
ENCODER 320
TRANSMIT PROCESS 396
301
RECEIVE FRONT END 380
A/D 392
FROM LINE 395
PRELIMINARY ECHO CANCELLER 390
BULK DELAY 382
AUTOMATIC GAIN CONTROL 389
POWER MEASUREMENT 388
NEAR-END ECHO INTERPOLATOR 381
FAR-END ECHO INTERPOLATOR 383
CLOCK RECOVERY 384
RECEIVE SIGNAL INTERPOLATOR 385
S SIGNAL DETECTOR 386
A/B SIGNAL DETECTOR 387
DEMODULATOR-1 371
MAIN NEAR-END ECHO CANCELLER 373
MAIN FAR-END ECHO CANCELLER 374
EQUALIZER 375
DECODER 360
RECEIVE PROCESS 397
DEMODULATOR 372
DEMODULATOR 370
302
FEEDBACK DATA PATH, COMPLEX SIGNAL
FEEDBACK DATA PATH, REAL SIGNAL
MAIN DATA PATH, COMPLEX SIGNAL
MAIN DATA PATH, REAL SIGNAL

FIG. 3

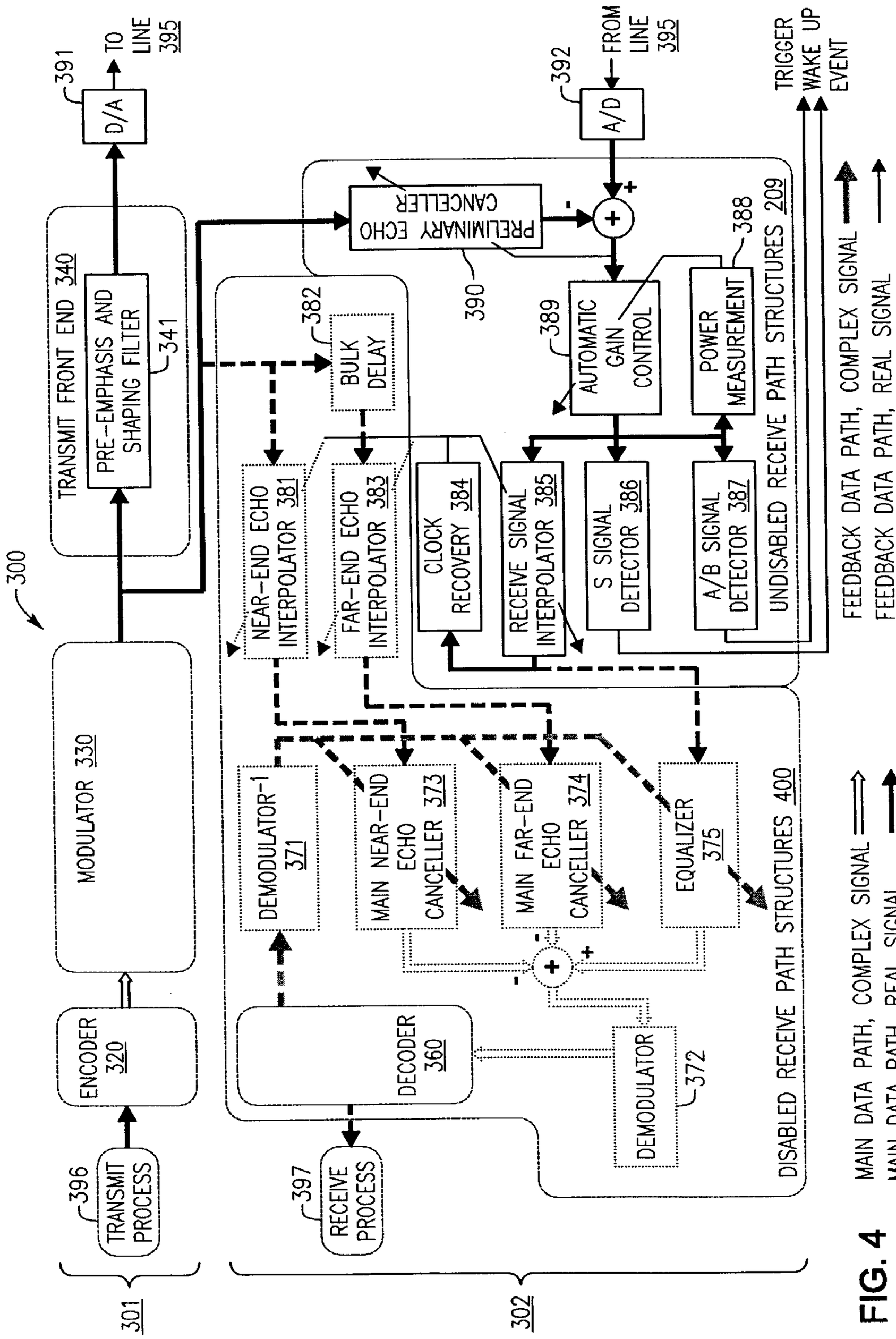

300
301
302
ENCODER 320
MODULATOR 330
TRANSMIT FRONT END 340
PRE-EMPHASIS AND SHAPING FILTER 341
D/A 391
TO LINE 395
TRANSMIT PROCESS 396
RECEIVE PROCESS 397
A/D 392
FROM LINE 395
PRELIMINARY ECHO CANCELLER 390
BULK DELAY 382
NEAR-END ECHO INTERPOLATOR 381
FAR-END ECHO INTERPOLATOR 383
CLOCK RECOVERY 384
RECEIVE SIGNAL INTERPOLATOR 385
S SIGNAL DETECTOR 386
A/B SIGNAL DETECTOR 387
AUTOMATIC GAIN CONTROL 389
POWER MEASUREMENT 388
UNDISABLED RECEIVE PATH STRUCTURES 209
TRIGGER WAKE UP EVENT
SIGNAL
DEMODULATOR-1 371
MAIN NEAR-END ECHO CANCELLER 373
MAIN FAR-END ECHO CANCELLER 374
EQUALIZER 375
DISABLED RECEIVE PATH STRUCTURES 400
DECODER 360
DEMODULATOR 372
FEEDBACK DATA PATH, COMPLEX SIGNAL
FEEDBACK DATA PATH, REAL SIGNAL
MAIN DATA PATH, COMPLEX SIGNAL
MAIN DATA PATH, REAL SIGNAL

FIG. 4

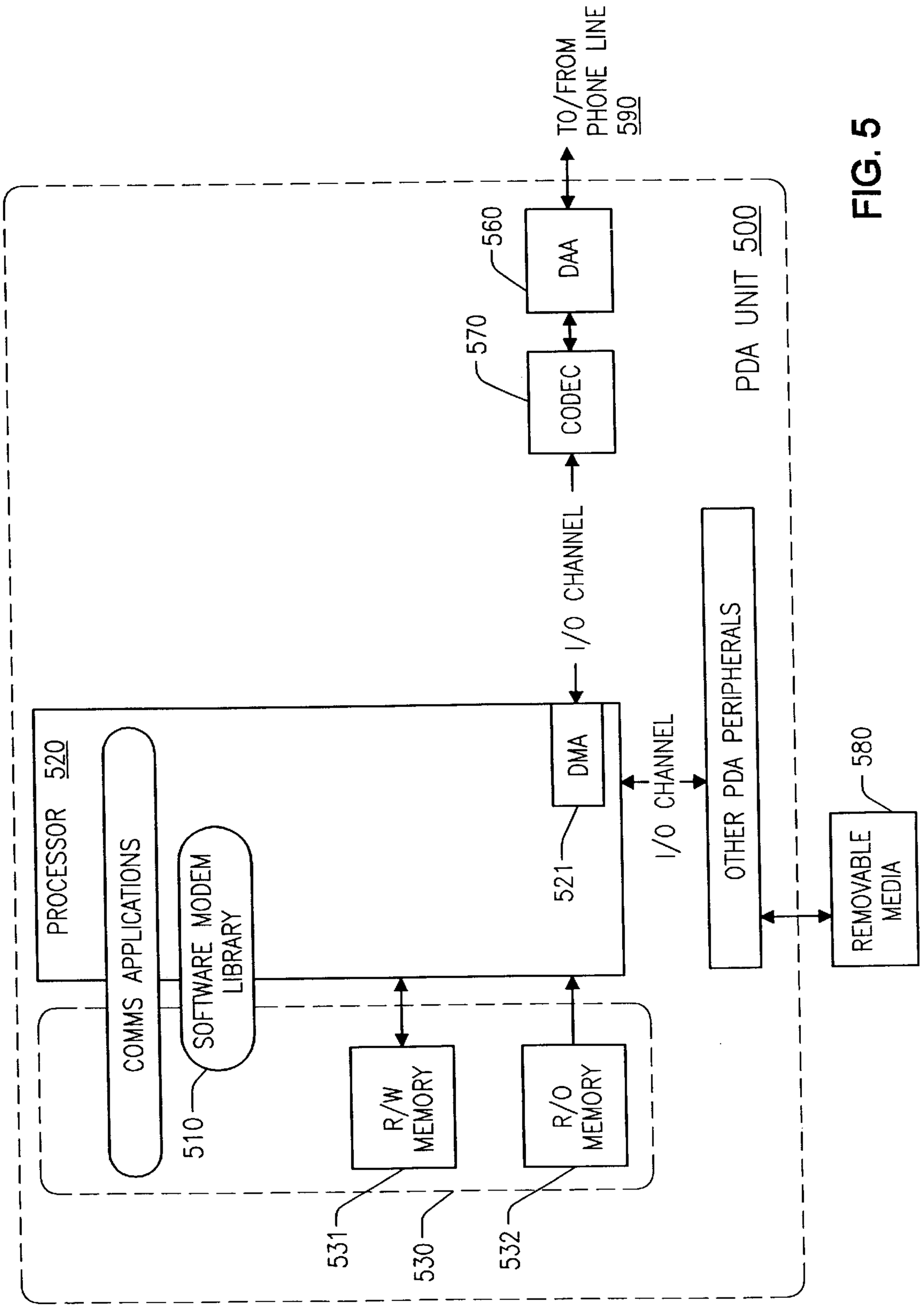

TO/FROM PHONE LINE 590
560
DAA
570
CODEC
I/O CHANNEL
PDA UNIT 500
PROCESSOR 520
COMMS APPLICATIONS
SOFTWARE MODEM LIBRARY
510
DMA
521
I/O CHANNEL
OTHER PDA PERIPHERALS
REMOVABLE MEDIA
580
R/W MEMORY
R/O MEMORY
531
530
532

FIG. 5

PROCESSOR SHARING TECHNIQUE FOR COMMUNICATIONS AND OTHER DATA PROCESSING ON A SAME PROCESSOR

This application is a Continuation of application Ser. No. 08/780,611, issued as U.S. Pat No. 5,995,540 filed Jan. 8, 1997, the entirety of which is incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates to modems, and in particular to systems and methods for reducing the processing requirements of a modem.

2. Description of the Related Art

Modems are communications devices which employ digital modulation techniques to transmit binary data over analog communications channels, e.g., telephone lines. Typically, two modems communicate over a single channel, with one modem at each end of the channel. Signal processing structures implemented at each modem provide encoding, modulation, filtering, interpolation, echo cancellation, signal detection, equalization, demodulation, and decoding functionality. Modems typically conform to international standards to ensure interoperability with modems from other manufacturers. One such standard is the V.34 specification described in V.34, *A Modem Operating at Data Signalling Rates of up to* 28 800*bits/s for Use on the General Switched Telephone Network and on Leased Point-to-Point* 2-*Wire Telephone-Type Circuits*, dated September, 1994 (previously CCITT Recommendation V.34), which is hereby incorporated herein, in its entirety, by reference.

Traditional modem implementations include one or more dedicated digital signal processors (or DSPs) on which signal processing algorithms execute during periods of modem operations. A computer system may incorporate such a modem implementation, and in addition, typically includes application and operating system software executable on a general purpose processor (e.g., any of a variety of processors including MIPS™ R3000, R4000, and R5000 processors, processors conforming to the SPARC™, PowerPC™, Alpha™, PA-RISC™, or x86 processor architectures, etc.). Software executing on the general purpose processor sends and receives data via the modem implementation using input/output (I/O) ports, direct memory access (DMA), or other I/O structures and methods suitable for a particular general purpose processor and operating system combination.

Since a typical modem implementation includes a dedicated DSP not shared with other signal processing functions of a larger computer system, the modem's DSP and the signal processing algorithms designed to run thereon are selected and designed to meet the peak computation load of the modem. DSP cycles are either used or lost. For this reason, signal processing algorithms implementing the complete steady-state functionality of modem transmit and receive paths are typically executed on a DSP at full speed for the duration of a modem connection.

For many portable device applications such as Personal Digital Assistants (PDAs), portable computers, and cellular phones, power consumption, battery life, and overall mass are important design figures of merit. In addition, very small part counts are desirable for extremely-small, low-cost consumer devices. Modem communications are desirable in many such portable device applications. However, traditional DSP implementations of the underlying signal processing capabilities create substantial power demands, require increased part counts, and because of the power consumption of a discrete DSP, typically require larger, heavier batteries.

A modem implemented as software executable on a general purpose computer could reduce part counts, power demands, and overall mass of a computer system by eliminating the DSP, its power consumption, and some of the battery capacity otherwise required. However, to fully benefit from the elimination of a DSP, such a software modem needs to co-exist with existing operating systems and applications, running on the same general purpose processor as the operating systems and applications. Unfortunately, in a computer system which includes such a software modem, the load on the general purpose processor can be significant, slowing operating system and application programs, even when the software modem is not sending or receiving data. For example, FIG. **1** depicts a 14 MIPS steady-state computational load, which is exemplary of a software implementation of a traditional modem maintaining an active connection with both transmitter and receiver structures active for the duration of the connection.

SUMMARY OF THE INVENTION

The computational load imposed by communications software executed on a processor can be significantly reduced by exploiting periods during an active connection when no data is being received. In particular, execution of many receive path signal processing algorithms can be disabled when no data is being received. The transmit path continues output modulation as with a normal connection so as to trick a remote communications device into believing the connection is still normal. However, because substantial portions of the local receive path can be disabled, computational load on the processor is reduced and additional compute cycles are freed for application and/or operating system program use.

Disabled portions of the receive path are re-enabled (restarted) in response to wake up events. For example, receive path algorithms can be re-enabled under program control, e.g., when the user, application program, or operating system initiates data transmission, such as in response to a user selection of a new Universal Resource Locator (URL) within a World-Wide Web (WWW) browser application. Because many network-based applications, e.g., browsers, electronic mail clients and servers, message retrieval clients, netcasting receivers, etc., exploit a transactionbased model of interprocess communication, receive path algorithms can be enabled coincident with the start of such a transaction and disabled coincident with completion of the transaction. Receive path algorithms can also be periodically re-enabled, e.g., at programmable intervals. In this way, the receive path is periodically available for retransmission of data missed while the receive path algorithms were disabled (i.e., while the receiver was "asleep," or in "doze" mode). Higher level data protocols (e.g., TCP/IP) will retransmit any data that was missed while the receiver was "asleep."

In addition to program controlled and periodic wake up events, wake up events may be triggered by the remote communications device. For example, in a modem embodiment in accordance with ITU-T Recommendation V.34, remote retrain, remote rate renegotiation, and remote clear-down requests are received by an undisabled set of receive path signal processing algorithms. A wake up event is triggered in response to detection of a remote retrain, remote

rate renegotiation, or remote cleardown request and disabled portions of the local modem's receive path are re-enabled in response thereto.

By providing functionality (e.g., in the form of functions, procedures, and/or methods) for enabling and disabling portions of a software modem's receive path algorithms, a software modem application program interface (API) in accordance with an embodiment of the present invention allows application-and operating system processes to switch between a doze mode (with low computational load) during idle receive time and a full operational mode (with full computational load) during active receive time. In this way, compute cycles otherwise allocated to idle-time execution of receive path algorithms are allocable to application program or operating system processes. In an exemplary implementation, computational requirements for the software modem can be reduced from 100% in steady-state, to less than 25% of the steady state load during idle receive time, i.e., when no data is being received by the software modem. In contrast, traditional approaches require the full 100% whether or not the software modem is receiving data.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention may be better understood, and its numerous objects, features, and advantages made apparent to those skilled in the art by referencing the accompanying drawings.

FIG. **1** is a state transition diagram depicting illustrative transitions and associated computational loads for a traditional hardware modem or software modem.

FIG. **2** is a state transition diagram depicting illustrative transitions and associated computational loads for a software modem providing reduced processing requirements during idle receive time in accordance with an exemplary embodiment of the present invention.

FIG. **3** is a block diagram depicting functional modules and data flows for an illustrative embodiment of a modem in accordance with an exemplary embodiment of the present invention.

FIG. **4** is a block diagram depicting functional modules and data flows for the modem of FIG. **3**, wherein an illustrative portion of the receive path structures thereof are disabled for reduction of processing requirements during idle receive time in accordance with an exemplary embodiment of the present invention.

FIG. **5** is a block diagram of an exemplary Personal Digital Assistant (PDA) system embodiment including a general purpose processor, registers, and memory for executing a software implementation of a modem such as that depicted in FIGS. **3** and **4**.

The use of the same reference symbols in different drawings indicates similar or identical items.

DESCRIPTION OF THE PREFERRED EMBODIMENT(S)

FIG. **2** depicts modem operating state transitions for an exemplary modem. Upon connection (with a second remote modem), the modem transitions from inactive state **201** to a steady state communications state **202** during which transmit and receive path structures of the modem are active. Steady state communications impose a significant computational load, illustratively 14 MIPS, which in an embodiment of the modem as software executable on a general purpose processor, depletes the number of general purpose processor cycles available for application and operating system software. On disconnect, the modem transitions from steady state communications state **202** to inactive state **203**.

A reduction in processing requirements during idle receive time is provided as follows. On either a doze requestor idle timeout event, the modem transitions from steady state communications state **202** to a doze state **204** during which transmit structures of the modem are active, but during which a substantial portion of receive path structures are disabled. Particular receive path structures are disabled and those which remain enabled are described below with reference to FIGS. **3** and **4**. A doze request may be triggered by software (e.g., application or operating system program **206**) interacting with the modem, for example, by a browser application which has successfully received (or downloaded) the data associated with a user selected universal resource locator (URL). Alternatively, software associated with an application, an operating system, or an application program interface (API) for the modem itself, may monitor modem receive activity and trigger an idle timeout transition to doze state **204** after a timeout interval. Illustratively, FIG. **2** depicts an idle timeout triggered by timer **207**. Alternative modem embodiments may support either or both of the doze request and idle timeout transitions from steady state communications state **202** to doze state **204**.

Once in doze state **204**, the modem imposes a substantially reduced computational load, illustratively, less than 4 MIPS. The modem remains in doze state **204** until a wake up event occurs. Such a wake up event can be generated locally or received from the remote modem. Regarding locally generated wake up events, doze state **204** can be exited under program control, e.g., coincident with a send_data ( ) call by application or operating system program **206**, such as in response to a user selection of a new Universal Resource Locator (URL) within a World-Wide Web (WWW) browser application. Because many network-based applications, e.g., browsers, electronic mail clients and servers, message retrieval clients, netcasting receivers, etc., exploit a transaction-based model of interprocess communication, the modem transitions under program control from doze state **204** to rate renegotiation state **205** coincident with the start of such a transaction and transitions to doze state **204** coincident with completion of the transaction. In this way, receive path structures can be enabled coincident with the start of such a transaction and disabled coincident with completion of the transaction.

In addition to a locally-generated, application-triggered wake up event, the modem also transitions from doze state **204** to rate renegotiation state **205** in response to a periodic wake up event generated by timer **211**. In this way, receive path structures are periodically re-enabled, e.g., at programmable intervals. Because data missed while portions of the receive path were disabled (i.e., while the software modem receiver was in doze state **204**) will be re-sent by higher level data protocols (e.g., protocols above the physical, or modulation, layer that guarantee in-order, error-free delivery such as TCP/IP, the Appletalk® protocol, IPX dialup, ZMODEM, etc.), periodic re-enabling of the receive path structures allows for receipt of previously-missed data when re-sent by the higher level data protocols (and therefore retransmitted by the remote modem). Suitable wake-up intervals will be protocol specific; however, a typical wake-up interval (and corresponding period of reduced computational load) is approximately 10 seconds. Alternative embodiments, such as a modem embodiment which does not perform an idle timeout transition from steady state communications state **202** to doze state **204**, may eliminate the

periodic wake up if the protocols used by a particular instance of application or operating system program **206**, e.g., post office protocol (POP**3**) used by an electronic mail client, hypertext transfer protocol (HTTP) used by a browser application, etc., are polling or request/response-type protocols, assuming that other layers of the protocol stack (e.g., link-layer protocols) do not require wake-up.

Wake up events can also be generated in response to transmissions from the remote modem. Although a substantial portion of receive path structures are disabled in doze state **204**, an undisabled set of receive-path structures **209** remain enabled during doze state **204** (see description below in the context of FIGS. **3** and **4**) so as to detect requests from the remote modem. In an exemplary V.34 embodiment, undisabled set of receive-path structures **209** allows the modem to detect remote retrain, remote rate renegotiation, and remote cleardown requests from remote modem **208** as described in ITU-T Recommendation V.34, *A Modem Operating at Data Signalling Rates of up to* 28 800*bits/s for Use on the General Switched Telephone Network and on Leased Point-to-Point* 2-*Wire Telephone-Type Circuits*, dated September, 1994 (hereafter ITU-T Recommendation V.34) §§ 11.5–11.7 of which are incorporated herein by reference. In response to such detections, a remote wake up event is triggered and the modem transitions from doze state **204** to corresponding rate renegotiation, retrain, and clear down states **205**, **210** and **212**.

In an exemplary V.34 embodiment, rate renegotiation is used to "wake-up" the receiver. Whether the modem is "waked-up" from doze state **204** in response to a locally-generated, application-triggered wake up event, in response to a periodic wake up event generated by timer **207**, or in response to a remote rate renegotiation wake up event, rate renegotiation is performed at rate renegotiation state **205** in accordance with the requirements of ITU-T Recommendation V.34, § 11.6 of which is incorporated herein by reference. Thereafter, the modem transitions from rate renegotiation state **205** back to steady state communications state **202** with re-enabled receive path structures. Rate renegotiation serves to retrain receive path structures, particularly demodulator far-end phase locked loop structures, if any, which have become stale since the modem's transition to doze state **204**. Alternative embodiments may perform resynchronization, e.g., data driven reacquisition, instead of rate renegotiation in response to a locally-generated, application-triggered wake up event (e.g., from application or operating system program **206**) or in response to a periodic wake up event generated by timer **207**.

Retrain event handling is analogous. Retraining is performed at retrain state **210** in accordance with the requirements of ITU-T Recommendation V.34, § 11.5 of which is incorporated herein by reference. Thereafter, the modem transitions from retrain state **210** back to steady state communications state **202** with re-enabled receive path structures. Rate renegotiation serves to retrain receive path structures, particularly demodulator far-end phase locked loop structures, if any, which have become stale since the modem's transition to doze state **204**.

FIGS. **3** and **4** depict transmit path structures **301** and receive path structures **302** for an exemplary V.34 modem **300** embodiment. Transmit path structures **301** and receive path structures **302** include fixed and adaptive filter implementations and other signal processing structures for modulation and demodulation of signals in accordance with the signaling requirements of ITU-T Recommendation V.34. In the exemplary embodiment of FIGS. **3** and **4**, filter implementations and other signal processing structures are implemented as software executable on a general purpose processor and enabling and disabling of portions of the receive path is accomplished in software. Persons of ordinary skill in the art will recognize a wide variety of methods for predicating execution of receive path code on modem state. FIG. **3** depicts receive path structures **302** in accordance with steady state communications state **202**, while FIG. **4** depicts receive path structures **302** in accordance with doze state **204** (including disabled receive path structures **400**).

Transmit path structures **301** include encoder **320**, modulator **330**, and pre-emphasis and shaping filter **341**. Receive path structures **302** include, decoder **360**, demodulation and channel impairment compensation module **370**, and receive front end module **380**. Although the structures of FIGS. **3** and **4** may be suitably implemented in custom circuitry, using a programmed custom (or commercially-available) DSP, as software executable on a general purpose processor, or as any combination of the above, and although doze state **204** can be provided in any of such implementations, the freeing (during doze state **204**) of general purpose processor cycles for use by application and/or operating system software is an advantage of an embodiment of V.34 modem **300** primarily or entirely implemented as software executable on a general purpose processor.

Persons of ordinary skill in the art will recognize a variety of suitable software implementations for structures along transmit and receive data paths, including algorithms for both performing the signal processing functions defined by the structures and for adaptively updating the structure definitions, e.g., by adaptively updating filter coefficients. The particular structures depicted in FIGS. **3** and **4** are merely illustrative of an exemplary set of suitable implementations. Alternative embodiments may incorporate transmit and receive path structures of any suitable design and persons of ordinary skill in the art will recognize suitable sets of receive path structures for enabling and disabling as now described with reference to the exemplary embodiment of V.34 modem **300**.

Referring now to the receive data path of V.34 modem **300**, receive path structures **302** (i.e., software implementations thereof) for receive front end module **380**, demodulation and channel impairment compensation module **370**, and decoder **360** are all active (enabled) while V.34 modem **300** is operating in steady state communications state **202**. In contrast, and as shown in FIG. **4**, a substantial portion of receive path structures **302** are disabled while V.34 modem **300** is operating in doze state **204**. The set of disabled receive path structures **400** is exemplary and implementations of V.34 modem **300** wherein larger, smaller, or different sets of receive path structures are disabled during doze state **204** are also suitable. For example, individual receive path structures disabled during doze state **204** in the embodiment of FIG. **4**, may instead remain enabled in alternative embodiments, though at the cost of additional computational load. Furthermore, receive path structures which remain enabled during doze state **204** in the embodiment of FIG. **4**, may be disabled in alternative embodiments. For example, clock recovery module **384** and receive signal interpolator **385** may be disabled during doze state **204**, though clock phase will need to be reacquired on transitions from doze state **204** to steady state communications state **202** via rate renegotiation state **205** or retrain state **210**. Some or all of S signal detector **386**, A/B signal detector **387**, automatic gain control **389**, power measurement module **388**, and preliminary echo canceller **390** may be disabled during doze state **204**, though degrading or eliminating the ability of V.34 modem **300** to detect remote wake up events (e.g., remote

retrain, remote rate renegotiation, and remote cleardown) and appropriately respond thereto.

Receive front end module **380** receives the output of the A/D converter **392** as an input. A/D converter **392** couples to transmission line **395**. Preliminary echo canceller **390** is implemented as a real data/real coefficients adaptive filter using any suitable filter implementation. In the steady state and doze state embodiments of FIGS. **3** and **4**, preliminary echo canceller **390** receives as an input a white signal from the output of the modulator **330**. Preliminary echo canceller **390** uses a stochastic gradient updating algorithm for adaptation during half duplex of V.34 training and is not updated during data mode. This preliminary stage of echo cancellation reduces echo level relative to the receive signal level so that subsequent stages such as clock recovery, signal detection, and automatic gain control (each not shown) will not be significantly affected by the echo. Preliminary echo canceller **390** remains enabled during doze state **204**, allowing S signal detector **386** and A/B signal detector **387** to detect remote retrain, remote rate renegotiation, and remote cleardown requests from the remote modem during doze state **204** and trigger wake up events in correspondence therewith. Alternative embodiments may provide distinct near- and far-end preliminary echo canceller structures. S signal detector **386** is employed to detect S-to-$\overline{S}$ transitions indicative of rate renegotiation and cleardown requests as described in ITU-T Recommendation V.34, §§ 11.6–7 of which are hereby incorporated by reference. Similarly, A/B signal detector **387** is employed to detect tone A (if V.34 modem **300** is the call modem) or tone B (if V.34 modem **300** is the answer modem) indicative of retrain requests as described in ITU-T Recommendation V.34, § 11.5 of which is hereby incorporated by reference.

Receive path structures **302** implemented along the receive data path should be synchronized with the remote modem signal. In the exemplary embodiment of FIGS. **3** and **4**, an adaptive FIR filter is used to perform the interpolation. Adaptive FIR filters implemented in any suitable manner (including DSP and general purpose processor alternatives described above) are used to interpolate the receive signal (at receive signal interpolator **385**) as well as to interpolate delayed and undelayed versions of the modulator output (at far-end echo interpolator **383** and near-end echo interpolator **381**) used as inputs for corresponding far- and near-end main echo cancellers **373** and **374**. The filter coefficients are adjusted based on timing phase and frequency recovered from the remote modem signal by clock recovery module **384**. The adaptation algorithm is performed by a two-stage combination of a poly-phase filter and linear interpolations. Persons of ordinary skill in the art will appreciate a variety of suitable implementations of poly-phase filters, as well as alternative adaptation algorithms. In the exemplary embodiment of FIG. **4**, both clock recovery module **384** and receive signal interpolator **385** are included in the undisabled set of receive-path structures **209**. In this way, synchronization with the remote mode clock is maintained even in doze mode. However, alternative embodiments may optionally include clock recovery module **384** and receive signal interpolator **385** in the set of disabled receive path structures **400**, though clock phase will need to be reacquired on transitions from doze state **204** to steady state communications state **204** via rate renegotiation state **205** or retrain state **210**.

Demodulator **372**, a corresponding inverse structure (demodulator^{-1} **371**), and decoder **360** provide a feedback loop for adaptive updates to the coefficients defining main near-end echo canceller **373**, main far-end echo canceller **374**, and equalizer **375**. V.34 modem **300** may optionally include a phase locked loop to compensate for frequency offset and phase jitter on transmission line **395**. Regarding demodulation and channel impairment compensation module **370**, a variety of alternative echo canceller and equalizer configurations are suitable. Several such configurations are described in greater detail in U.S. Pat. No. 5,864,545 entitled, "System and Method for improving Convergence During Modem Training and Reducing Computational Load During Steady-State Modem Operations," naming Gonikberg and Liang as inventors and filed on Dec. 6, 1996, the entirety of which is hereby incorporated by reference.

Decoder **360** converts the demodulated complex symbols into a bit stream which is supplied to receiver process **397**. Transmit process **396** and receiver process **397** may be the same process. Decoder **360** performs nonlinear decoding, linear prediction, trellis decoding, constellation decoding, shell demapping, and data de-framing, all as described in respective sections of the V.34 recommendation, which is incorporated herein by reference. Persons of ordinary skill in the art will recognize a variety of alternative implementations of decoder **360** in accordance with the requirements of the V.34 recommendation. In addition, persons of ordinary skill in the art will recognize a variety of alternative configurations of decoder **360** suitable to modem implementations in accordance with other communications standards such as V.32, V.32bis, etc.

Referring now to the transmit data path of V.34 modem **300**, transmit process **396** supplies a bit stream to a V.34 implementation of encoder **320**. Encoder **320** converts the input bit stream into a baseband sequence of complex symbols which is used as input to modulator **330**. Encoder **320** performs shell mapping, differential encoding, constellation mapping, preceding and 4D trellis encoding, and nonlinear encoding, all as described in respective sections of ITU-T Recommendation V.34, §§ 9.1–9.7 of which are hereby incorporated by reference.

Modulator **330** converts the baseband sequence of complex symbols from the output of the encoder into a passband sequence of real samples. In particular, modulator **330**:

1. multiplies the complex baseband sequence by the carrier frequency; and
2. converts the complex signal to real.

If the spectrum of the modulator output is sufficiently white, it can be used as an input to receiver echo cancellers, as described below.

Shaping and pre-emphasis filter **341** provides square-root-of-raised-cosine shaping as well as pre-emphasis filtering specified by section 5.4 of the V.34 recommendation, which is incorporated herein by reference. Square-root raised cosine complex shaping and pre-emphasis filtering are implemented using any suitable filter implementation. The output of shaping and pre-emphasis filter **341** is an output of the transmitter portion of V.34 modem **300** is provided to D/A converter **391**. D/A converter **391** couples to transmission line **395**.

Transitions between steady state communications state **202** and doze state **204** are performed as described below. It is important that the remote modem not go into a hang-up state, because re-dialing is very time consuming. To put the modem in doze state **204**, for example, when no TCP/IP sessions are open and there is no data to be received, while avoiding a remote modem hang up:

1. application or operating system program **206** makes a doze request using StartModemDoze ( ) function defined by an application program interface (API), for example, by a software modem API;

2. the modem transitions to doze state **204** by setting a flag to signal transition to doze state **204**, and
3. in response to the set state of the flag,
   a) disabling the echo canceller input interpolation, i.e., far-end echo interpolator **383** and near-end echo interpolator **381**, and
   b) placing demodulation and channel impairment compensation module **370** in idle mode (during which no more symbols will be demodulated).

In an exemplary embodiment, decoder module **360** and receive data module **350** are also not executed during doze state **204** because data flows from idled demodulation and channel impairment compensation module **370** are halted. However, alternative embodiments may specifically disable or idle decoder module **360** and receive data module **350**.

To return from doze state **204** back to steady state communications state **202**, for example when the user or an application or operating system program requests data to be transmitted again, for example by the user selecting a URL to get a new web page, disabled receive path structures **400** are re-enabled as follows:

1. application or operating system program **206** makes a doze request using NoDoze ( ) function defined by an API,
2. the modem transitions from doze state **204** back to steady state communications state **202** via renegotiation state **205** by:
   a) re-enabling echo canceller input interpolation, i.e., far-end echo interpolator **383** and near-end echo interpolator **381**,
   b) placing the demodulation and channel impairment compensation module **370** in active mode, and
   c) initiating a rate renegotiation procedure with the remote modem in accordance with the requirements of ITU-T Recommendation V.34 to re-synchronize quickly.

Transitions from doze state **204** back to steady state communications state **202** which are triggered by a wake up event from the remote modem or by a local wake up analogously re-enable disabled receive path structures **400** by steps a, b, and c (above). In addition, transitions from doze state **204** back to steady state communications state **202** via retrain state **210** and triggered by a wake up event from the remote modem analogously re-enable disabled receive path structures **400** by steps a and b (above), together with step c' as follows:

c') initiating a retrain procedure with the remote modem in accordance with the requirements of ITU-T Recommendation V.34.

In an exemplary software embodiment, V.34 modem **300** includes an efficient general purpose processor implementation of an FIR filter as described in greater detail in U.S. patent application Ser. No.: 08/748,854 entitled "Efficient Implementation of An FIR Filter on a General Purpose Processor," naming Gonikberg and. Liang as inventors and filed Nov. 14, 1996, the entirety of which is hereby incorporated by reference. Along the transmit data path, pre-emphasis and shaping filter **341** is implemented using such an FIR filter implementation. Along the receive data path, echo interpolators **381** and **383**, receive signal interpolator **385**, preliminary echo canceller **390**, main near- and far-end echo cancellers **373** and **374**, and equalizer **375** are also implemented using such an FIR filter.

Other Embodiments

FIG. **5** depicts a Personal Digital Assistant (PDA) **500** incorporating a software modem library **510** including modules providing a software implementation of a V.34 modem **300**). In a such an embodiment, execution of code associated with an undisabled subset of receive path structures **302** is selectively enabled and disabled in accordance with transitions in V.34 modem **300** state from steady state communications state **202** to doze state **204** and back to steady state communications state **202**.

Input signal vectors and filter coefficient vectors suitable for providing the various filter implementations of interpolators, phase splitting filters, linear predictors, etc. (which have been described above with reference to FIGS. **3** and **4**) are loaded from memory **530** and output signal vectors are stored to memory **530**. In addition, executable instructions implementing the software modem library **510** (including implementations of transmit path structures **301** and receive path structures **302**) and suitable for execution on general purpose processor **520** are also stored in, and loaded from, memory **530**.

In an exemplary embodiment, general purpose processor **520** includes a MIPS R3000 RISC microprocessor, although a wide variety of alternative processor implementations are also suitable, including, for example, R4000 and R5000 processors, processors conforming to the Sparc, PowerPC, Alpha, PA-RISC, or x86 processor architectures, etc. General purpose processor **520** includes a DMA channel **521** for interfacing to telecommunication circuits (illustratively, phone line **590**) via codec **570** and Digital-to-Analog/Analog-to-Digital (DAA) converter **560**. Of course, memory **530** may include either read/write memory **531** or read/write memory **531** in combination with read-only memory **532**. Persons of ordinary skill in the art will recognize a variety of suitable allocations of code and data to each. Removable media **580** provides a mechanism for supplying the executable instructions implementing software modem library **510**.

While the invention has been described with reference to various embodiments, it will be understood that these embodiments are illustrative and that the scope of the invention is not limited to them. Many variations, modifications, additions, and improvements of the embodiments described are possible and may fall within the scope of the invention as defined by the claims which follow.

What is claimed is:

1. A method of utilizing a processor for both signal processing and applications program tasks, the method comprising:

concurrently executing the signal processing tasks and the applications program tasks on the processor, the signal processing tasks including both receive path tasks and transmit path tasks associated with an active communications session; and

freeing compute cycles on the processor for use by the applications program tasks by disabling a substantial portion of the receive path tasks while maintaining the active communications session.

2. The method of claim **1**,

wherein compute cycle freeing is performed during idle receive time of the active communications session.

3. The method of claim **1**, further comprising:

restoring, during the active communications session, the disabled portion of the receive path tasks.

4. The method of claim **1**, further comprising:

restoring the disabled portion of the receive path tasks in response to a wake up event.

5. The method of claim **1**,

wherein during the disabling of the portion of the receive path tasks, compute load associated with the signal

processing tasks is reduced to less than about 25% of a full load therefor, and the compute cycles for use by the applications program are increased correspondingly thereby.

6. The method of claim **1**,

wherein the compute cycle freeing is performed after transitioning from an inactive state to a steady-state operation of the active communications session.

7. A data communications apparatus comprising:

an adapter for interfacing a processor of a host computer to a communications channel, the adapter including an analog-to-digital converter coupled to convert an analog signal received from the communications channel to a series of digital samples; and

instructions adapted for execution by the processor, the instructions defining:

receive path tasks for operation on the digital samples; transmit path tasks; and

a utilization control routine which selectively disables a substantial portion of the receive path tasks in response to a triggering event,

wherein, during the disabling, compute cycles normally utilized by the disabled portion of the receive path tasks are freed for applications program tasks also executing on the processor.

8. The data communications apparatus of claim **7**,

wherein the triggering event includes at least one of an idle timeout and a doze request.

9. The data communications apparatus of claim **7**,

wherein the utilization control routine re-enables the disabled portion of the receive path tasks in response to a wake up event.

10. The data communications apparatus of claim **9**,

wherein the wake up event includes at least one of a local wake up event, a local periodic wake up and a remote wake up event, and

wherein an active communications session is maintained during the disabling.

11. The data communications apparatus of claim **7**,

wherein undisabled ones of the receive path tasks are maintained during the disabling to trigger a remote wake up event.

12. The data communications apparatus of claim **7**,

wherein the triggering event occurs after transitioning from an inactive state to a steady-state operation of the data communications apparatus.

* * * * *